AIAA JOURNAL
Vol. 30, No. 11, November 1992

Implicit Navier-Stokes Solver for Three-Dimensional Compressible Flows

Seokkwan Yoon* and Dochan Kwak†
NASA Ames Research Center, Moffett Field, California 94035

A three-dimensional numerical method based on the lower-upper symmetric-Gauss-Seidel implicit scheme in conjunction with the flux-limited dissipation model is developed for solving the compressible Navier-Stokes equations. A new computer code based on this method requires only 9 μs per grid point per iteration on a single processor of a Cray YMP computer and executes at the sustained rate of 175 MFLOPS. A reduction of three orders of magnitude in the residual for a high-Reynolds-number flow using 636K grid points is obtained in 38 min. The computational results compare well with available experimental data.

I. Introduction

SINCE the computational requirements for direct simulation of turbulent flows about complex three-dimensional geometries are still beyond the reach of the most powerful supercomputers, most numerical algorithms developed so far focus on the solution of the Reynolds-averaged Navier-Stokes equations, which can be obtained by ensemble averaging of rapidly fluctuating components. The governing equations of fluid flows can be integrated by either explicit or implicit methods. Although explicit schemes have been successful in solving the Euler equations for inviscid flows, the efficiency of explicit schemes in solving the Navier-Stokes equations is limited by the Courant-Friedrichs-Lewy condition, which is especially restrictive when the computational grid is highly clustered to resolve the viscous boundary layer. When the time step limit imposed by an explicit stability bound is significantly less than the accuracy requirement, implicit schemes are often preferred. However, the tradeoff between a decreased number of iterations and an increased operation count per iteration for the implicit methods must be considered. The fastest convergence rate may be attained by an unfactored implicit scheme that directly inverts a large block banded matrix using Gaussian elimination. Such a scheme is impractical in three dimensions because of the rapid increase of the number of operations as the number of mesh points increases and because of the large memory requirement.

Yoon and Jameson[1-3] introduced an implicit algorithm based on a lower-upper factorization and Gauss-Seidel relaxation for the Euler and Navier-Stokes equations. Since then, the lower-upper symmetric-Gauss-Seidel (LU-SGS) scheme has been successfully implemented by many researchers. Shuen and Yoon[4] applied the method to supersonic combustion ramjet problems for the National Aero-Space Plane to take advantage of the algorithm's property that reduces the size of the matrix for reacting flows with finite rate chemistry. The resulting computer program RPLUS was named after the original perfect gas code PLUS (Program using LU Schemes).[1] A variation of the PLUS code, IPLUS, was applied to internal flows through turbomachinery cascades in conjunction with an interactive grid generation technique by Choo et al.[5] Another variant, HPLUS, demonstrated the robustness of an LU scheme at high Mach numbers.[6] Rieger and Jameson[7] developed a three-dimensional code based on an early version of the PLUS code and applied it to Hermes, the European space shuttle. Yu et al.[8] extended the RPLUS code to three dimensions. Coirier[9] developed a finite difference version of the RPLUS code for corner and gap-seal calculations. However, the accuracy and efficiency of the preceding codes have been limited by the artificial viscosity model.[10]

Yoon and Kwak[11,12] proposed that a variety of schemes could be constructed in the framework of the LU-SGS algorithm by different choices of Jacobian matrices of flux vectors and numerical dissipation models. The computer code CENS2D (Compressible Euler and Navier-Stokes) was written to study the effects of different dissipation models. It was observed that the blended first- and third-order model was the least accurate, whereas the flux-difference-split upwind-biased model was not only the most expensive but the least robust when the grid lines were not aligned with strong bow shock waves. It was concluded in the study that the flux-limited dissipation model was a practical alternative to upwind schemes because of its robustness, efficiency, and accuracy for high-speed external flows. Recently, promising results were reported using upwind-biased and total variation diminishing schemes with the LU-SGS implicit scheme. These results include the findings of Obayashi[13] for underexpanded plumes, Chen et al.[14] for forward-flight rotor flow, Loh and Golafshani[15] for flows in hybrid rocket motors, Yungster[16] for shock wave and boundary-layer interactions, and Imlay and Eberhardt[17] for flows past the Aeroassist Flight Experiment vehicle. In the meantime, the CENS2D code has been extended by Park and Yoon[18-20] to compute thermochemical nonequilibrium in hypersonic external flows using a multiple temperature model.

Although conventional implicit methods often achieve fast convergence rates, they suffer from greater computer time per iteration than explicit methods. The LU-SGS implicit scheme offers a potential for very low computer time per iteration as well as fast convergence. High efficiency can be achieved by accomplishing the complete vectorizability of the algorithm on oblique planes of sweep in three dimensions.[21] It has been demonstrated that the LU-SGS scheme requires less computational work per iteration than most existing schemes on a Cray YMP supercomputer in the case of three-dimensional viscous incompressible flows. One of the objectives of the present work is to provide standard performance figures that the LU-SGS scheme can achieve for three-dimensional compressible flows in conjunction with the flux-limited dissipation model by developing a new testbed code named CENS3D.

Received May 10, 1991; presented as Paper 91-1555 at the AIAA 10th Computational Dynamics Conference, Honolulu, HI, June 24-27, 1991; revision received April 7, 1992; accepted for publication April 10, 1992.

*Applied Computational Fluids Branch, MS 202A-1. Senior Member AIAA.

†Applied Computational Fluids Branch, MS 202A-1. Associate Fellow AIAA.

II. Navier-Stokes Equations

Let t be the time; ρ p, and T the density, pressure, and temperature, respectively; u, v, and w the velocity components in Cartesian coordinates (x, y, z); $\hat{Q}$ the vector of conserved variables; $\hat{E}$, $\hat{F}$, and $\hat{G}$ be the convective flux vectors; and $\hat{E}_v$, $\hat{F}_v$, and $\hat{G}_v$ the flux vectors for the viscous terms. Then the three-dimensional Navier-Stokes equations in generalized curvilinear coordinates (ξ, η, ζ) can be written as

$$\partial_t \hat{Q} + \partial_\xi(\hat{E} - \hat{E}_v) + \partial_\eta(\hat{F} - \hat{F}_v) + \partial_\zeta(\hat{G} - \hat{G}_v) = 0 \tag{1}$$

The flux vectors for compressible and incompressible flows are different. The flux vectors for compressible flow are

$$\hat{Q} = h\begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \qquad \hat{E} = h\begin{bmatrix} \rho U \\ \rho U u + \xi_x p \\ \rho U v + \xi_y p \\ \rho U w + \xi_z p \\ U(e + p) \end{bmatrix}$$

$$\hat{F} = h\begin{bmatrix} \rho V \\ \rho V u + \eta_x p \\ \rho V v + \eta_y p \\ \rho V w + \eta_z p \\ V(e + p) \end{bmatrix}, \qquad \hat{G} = h\begin{bmatrix} \rho W \\ \rho W u + \zeta_x p \\ \rho W v + \zeta_y p \\ \rho W w + \zeta_z p \\ W(e + p) \end{bmatrix} \tag{2}$$

where e is the total energy. The contravariant velocity components U, V, and W are defined as

$$\begin{aligned} U &= \xi_x u + \xi_y v + \xi_z w \\ V &= \eta_x u + \eta_y v + \eta_z w \\ W &= \zeta_x u + \zeta_y v + \zeta_z w \end{aligned} \tag{3}$$

The equation of state is needed to complete the set of equations for compressible flow:

$$p = (\gamma - 1)[e - \tfrac{1}{2}\rho(u^2 + v^2 + w^2)] \tag{4}$$

where γ is the ratio of specific heats. In the following equation h is the determinant of the inverse of transformation Jacobian matrix:

$$h = \begin{vmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{vmatrix} \tag{5}$$

The flux vectors for incompressible flow can be written in a similar way if the pseudocompressibility formulation[21] is used. In a finite volume formulation, h is identical to the mesh cell volume. The viscous flux vectors are

$$\begin{aligned} \hat{E}_v &= h[\xi_x E_v + \xi_y F_v + \xi_z G_v] \\ \hat{F}_v &= h[\eta_x E_v + \eta_y F_v + \eta_z G_v] \\ \hat{G}_v &= h[\zeta_x E_v + \zeta_y F_v + \zeta_z G_v] \end{aligned} \tag{6}$$

Their Cartesian components are

$$E_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + k\partial_x T \end{bmatrix}$$

$$F_v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{yx} + v\tau_{yy} + w\tau_{yz} + k\partial_y T \end{bmatrix} \tag{7}$$

$$G_v = \begin{bmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ u\tau_{zx} + v\tau_{zy} + w\tau_{zz} + k\partial_z T \end{bmatrix}$$

where

$$\begin{aligned} \tau_{xx} &= 2\mu\partial_x u - \tfrac{2}{3}\mu(\partial_x u + \partial_y v + \partial_z w) \\ \tau_{yy} &= 2\mu\partial_y v - \tfrac{2}{3}\mu(\partial_x u + \partial_y v + \partial_z w) \\ \tau_{zz} &= 2\mu\partial_z w - \tfrac{2}{3}\mu(\partial_x u + \partial_y v + \partial_z w) \\ \tau_{xy} &= \tau_{yx} = \mu(\partial_y u + \partial_x v) \\ \tau_{xz} &= \tau_{zx} = \mu(\partial_z u + \partial_x w) \\ \tau_{yz} &= \tau_{zy} = \mu(\partial_z v + \partial_y w) \end{aligned} \tag{8}$$

where the coefficient of viscosity μ and the coefficient of thermal conductivity k are decomposed into laminar and turbulent contributions:

$$\mu = \mu_l + \mu_t \tag{9}$$

$$k = \frac{\gamma}{\gamma - 1}\left(\frac{\mu_l}{Pr_l} + \frac{\mu_t}{Pr_t}\right) \tag{10}$$

where Pr_l and Pr_t denote laminar and turbulent Prandtl numbers.

Whereas the Euler equations can be obtained by neglecting the viscous terms, the thin-layer Navier-Stokes equations can be obtained by retaining the viscous flux vector in the direction normal to body surfaces.

III. Implicit Methods

The governing equations are integrated in time for both steady and unsteady flow calculations. For a steady-state solution the use of a large time step leads to fast convergence. For a time-accurate solution it is desirable that the time step is determined by the physics rather than the numerics. An unfactored implicit scheme can be obtained from a nonlinear implicit scheme by linearizing the flux vectors about the previous time step and dropping terms of the second and higher order:

$$[I + \alpha\Delta t(D_\xi \hat{A} + D_\eta \hat{B} + D_\zeta \hat{C})]\delta\hat{Q} = -\Delta t\hat{R} \tag{11}$$

where $\hat{R}$ is the residual:

$$\hat{R} = D_\xi(\hat{E} - \hat{E}_v) + D_\eta(\hat{F} - \hat{F}_v) + D_\zeta(\hat{G} - \hat{G}_v) \tag{12}$$

and I is the identity matrix. The expression $\delta\hat{Q}$ is the correction $\hat{Q}^{n+1} - \hat{Q}^n$, where n denotes the time level. D_ξ, D_η, and D_ζ are difference operators that approximate ∂_ξ, ∂_η, and ∂_ζ. $\hat{A}$, $\hat{B}$, and $\hat{C}$ are the Jacobian matrices of the convective flux vectors:

$$\hat{A} = \frac{\partial\hat{E}}{\partial\hat{Q}}, \qquad \hat{B} = \frac{\partial\hat{F}}{\partial\hat{Q}}, \qquad \hat{C} = \frac{\partial\hat{G}}{\partial\hat{Q}} \tag{13}$$

For compressible flow,

$$\hat{A} = \begin{bmatrix} 0 & \xi_x \\ \xi_x\tilde{q} - Uu & U - \xi_x u(\gamma-2) \\ \xi_y\tilde{q} - Uv & \xi_x v - \xi_y u(\gamma-1) \\ \xi_z\tilde{q} - Uw & \xi_x w - \xi_z u(\gamma-1) \\ U(\tilde{q} - \tilde{h}) & \xi_x\tilde{h} - Uu(\gamma-1) \end{bmatrix}$$

$$\times \begin{bmatrix} \xi_y & \xi_z & 0 \\ \xi_y u - \xi_x v(\gamma-1) & \xi_z u - \xi_x w(\gamma-1) & \xi_x(\gamma-1) \\ U - \xi_y v(\gamma-2) & \xi_z v - \xi_y w(\gamma-1) & \xi_y(\gamma-1) \\ \xi_y w - \xi_z v(\gamma-1) & U - \xi_z w(\gamma-2) & \xi_z(\gamma-1) \\ \xi_y\tilde{h} - Uv(\gamma-1) & \xi_z\tilde{h} - Uw(\gamma-1) & U\gamma \end{bmatrix}$$

where

$$\tilde{q} = \frac{\gamma-1}{2}(u^2 + v^2 + w^2) \tag{14}$$

$$\tilde{h} = (e + p)/\rho \tag{15}$$

The matrices $\hat{B}$ and $\hat{C}$ are similarly derived. Although the direct inversion method seems to be competitive with approximate factorization methods in the overall computing time in two dimensions,[22] direct inversion of a large block banded matrix of the unfactored scheme (11) appears to be impractical in three dimensions, as stated earlier.

To alleviate this difficulty, many investigators have focused on indirect methods. The popular alternating direction implicit (ADI) scheme by Beam and Warming[23] or Briley and McDonald[24] replaces the implicit operator of the unfactored scheme by a product of three one-dimensional operators:

$$(I + \alpha\Delta t D_\xi \hat{A})(I + \alpha\Delta t D_\eta \hat{B})(I + \alpha\Delta t D_\zeta \hat{C})\delta\hat{Q} = -\Delta t\hat{R} \tag{16}$$

The ADI scheme, which is unconditionally stable in two dimensions, becomes unstable in three dimensions, although numerical dissipation conditionally stabilizes the method. Because of three factors, the ADI scheme also introduces the error terms of $(\Delta t)^3$. The large factorization error associated with this scheme further reduces the rate of convergence. Despite these drawbacks, the ADI scheme has been successful due to the reduction of cost by the diagonalization of Jacobian matrices by Pulliam and Chaussee.[25] Obayashi and Kuwahara[26] developed a scheme by replacing each factor with bidiagonal LU factors:

$$(I + \alpha\Delta t D_\xi^- \hat{A}^+)(I + \alpha\Delta t D_\xi^+ \hat{A}^-)(I + \alpha\Delta t D_\eta^- \hat{B}^+)$$
$$\times (I + \alpha\Delta t D_\eta^+ \hat{B}^-)(I + \alpha\Delta t D_\zeta^- \hat{C}^+)(I + \alpha\Delta t D_\zeta^+ \hat{C}^-)\delta\hat{Q}$$
$$= -\Delta t\hat{R} \tag{17}$$

Stability and convergence characteristics of the LU-ADI scheme appear to be similar to the ADI scheme.

The factorization errors of two-factor schemes, which are of order $(\Delta t)^2$, are lower than the ADI scheme. Two-factor schemes can also be stable in three dimensions. Steger proposed a two-factor scheme[27,28] by partially splitting the flux vectors:

$$[I + \alpha\Delta t(D_\xi^- \hat{A}^+ + D_\eta\hat{B})][I + \alpha\Delta t(D_\xi^+ \hat{A}^- + D_\zeta\hat{C})]\delta\hat{Q}$$
$$= -\Delta t(D_\xi^- \hat{E}^+ + D_\xi^+ \hat{E}^- + D_\eta\hat{F} + D_\zeta\hat{G}) \tag{18}$$

The scheme was incorporated in the F3D code[28,29] and the CNS code.[30] The partially flux-split scheme is more expensive than the diagonalized ADI scheme because of block tridiagonal inversions.

An alternative two-factor scheme is based on an lower-upper (LU) factorization proposed by Steger and Warming[27] and Jameson and Turkel[31]:

$$LU\delta\hat{Q} = -\Delta t\hat{R} \tag{19}$$

where

$$L = I + \alpha\Delta t(D_\xi^- \hat{A}^+ + D_\eta^- \hat{B}^+ + D_\zeta^- \hat{C}^+)$$
$$U = I + \alpha\Delta t(D_\xi^+ \hat{A}^- + D_\eta^+ \hat{B}^- + D_\zeta^+ \hat{C}^-) \tag{20}$$

where D_ξ^-, D_η^-, and D_ζ^- are backward difference operators, and D_ξ^+, D_η^+, and D_ζ^+ are forward difference operators. Despite its early introduction in the late 1970s, the LU scheme had not been used until it was independently implemented by Buning and Steger,[32] Whitfield,[33] Buratynski and Caughey,[34] and Jameson and Yoon.[2,3] The cost of the LU scheme is more expensive than the diagonalized ADI scheme because of block diagonal inversions.

MacCormack[35] introduced an implicit line relaxation method based on back-and-forth symmetric sweeps in conjunction wtih upwind flux splitting. Although the line Gauss-Seidel relaxation method allowed significant increase of work per iteration compared to approximate factorization schemes due to multiple block tridiagonal inversions and sequential operations, it achieved very fast convergence rates. In fact, all of the implicit schemes mentioned earlier require much larger computational work per iteration than explicit schemes.

Yoon and Jameson[1] derived a new implicit algorithm by combining the advantages of LU factorization and SGS relaxation. The LU-SGS scheme has quite different L and U factors from those of the LU scheme. Unlike the line SGS relaxation scheme, no additional relaxation or factorization is required on planes of sweep. The LU-SGS scheme can be written as

$$LD^{-1}U\delta\hat{Q} = -\Delta t\hat{R} \tag{21}$$

where

$$L = I + \alpha\Delta t(D_\xi^- \hat{A}^+ + D_\eta^- \hat{B}^+ + D_\zeta^- \hat{C}^+ - \hat{A}^- - \hat{B}^- - \hat{C}^-)$$
$$D = I + \alpha\Delta t(\hat{A}^+ - \hat{A}^- + \hat{B}^+ - \hat{B}^- + \hat{C}^+ - \hat{C}^-) \tag{22}$$
$$U = I + \alpha\Delta t(D_\xi^+ \hat{A}^- + D_\eta^+ \hat{B}^- + D_\zeta^+ \hat{C}^- + \hat{A}^+ + \hat{B}^+ + \hat{C}^+)$$

In the framework of the LU-SGS algorithm, a variety of schemes can be developed by different choices of numerical dissipation models and Jacobian matrices of the flux vectors.[11] The matrix should be diagonally dominant to ensure convergence to a steady state. Jacobian matrices leading to diagonal dominance are constructed so that + matrices have nonnegative eigenvalues, whereas − matrices have nonpositive eigenvalues. For example,

$$\hat{A}^\pm = \hat{T}_\xi \Lambda_\xi^\pm \hat{T}_\xi^{-1}$$
$$\hat{B}^\pm = \hat{T}_\eta \Lambda_\eta^\pm \hat{T}_\eta^{-1} \tag{23}$$
$$\hat{C}^\pm = \hat{T}_\zeta \Lambda_\zeta^\pm \hat{T}_\zeta^{-1}$$

where $\hat{T}_\xi$ and $\hat{T}_\xi^{-1}$ are similarity transformation matrices of the eigenvectors $\hat{A}$. Another possibility is to construct Jacobian matrices of the flux vectors approximately to yield diagonal dominance:

$$\hat{A}^\pm = \tfrac{1}{2}[\hat{A} \pm \tilde{\rho}(\hat{A})I]$$
$$\hat{B}^\pm = \tfrac{1}{2}[\hat{B} \pm \tilde{\rho}(\hat{B})I] \tag{24}$$
$$\hat{C}^\pm = \tfrac{1}{2}[\hat{C} \pm \tilde{\rho}(\hat{C})I]$$

where, for example,

$$\tilde{\rho}(\hat{A}) = \kappa \max[|\lambda(\hat{A})|] \tag{25}$$

where $\lambda(\hat{A})$ represents eigenvalues of the Jacobian matrix $\hat{A}$, and κ is a constant that is ≥ 1. Stability and convergence can be controlled by adjusting κ either manually or automatically as the flowfield develops. The diagonal matrix of eigenvalues is

$$\hat{\Lambda}(\hat{A}) = \begin{bmatrix} U & 0 & 0 & 0 & 0 \\ 0 & U & 0 & 0 & 0 \\ 0 & 0 & U & 0 & 0 \\ 0 & 0 & 0 & U + C_\xi & 0 \\ 0 & 0 & 0 & 0 & U - C_\xi \end{bmatrix} \tag{26}$$

and

$$C_\xi = c\sqrt{\xi_x^2 + \xi_y^2 + \xi_z^2} \tag{27}$$

where c is the speed of sound:

$$c = \sqrt{\frac{\gamma p}{\rho}} \tag{28}$$

In the early days of development of codes such as the PLUS series, Eq. (21) was inverted in three steps:

$$\begin{aligned} \delta\hat{Q}^* &= -\Delta t D\hat{R} \\ \delta\hat{Q}^{**} &= L^{-1}\hat{Q}^* \\ \delta\hat{Q} &= U^{-1}\hat{Q}^{**} \end{aligned} \tag{29}$$

This is not a mathematically correct procedure, although no difference in the solution or convergence has been observed when D is a scalar diagonal matrix. The correct order used in INS3D-LU and CENS3D codes is

$$\begin{aligned} \delta\hat{Q}^* &= -\Delta t L^{-1}\hat{R} \\ \delta\hat{Q}^{**} &= D\hat{Q}^* \\ \delta\hat{Q} &= U^{-1}\hat{Q}^{**} \end{aligned} \tag{30}$$

It is interesting to note that the need for block inversions along the diagonals can be eliminated if we use the approximate Jacobian matrices of Eq. (24). Setting $\alpha = 1$ and $\Delta t = \infty$ yields a Newton-like iteration. Although a quadratic convergence of the Newton method cannot be achieved because of the approximate factorization, a linear convergence can be demonstrated. That is why the term *Newton-like* instead of *Newton* is used to distinguish the differences. The use of Newton-like iteration offers a practical advantage in that one does not have to find an optimal Courant number or time step to reduce the overall computer time. If two-point one-sided differences are used, Eq. (22) reduces to

$$\begin{aligned} L &= \tilde{\rho}I - \hat{A}^+_{i-1,j,k} - \hat{B}^+_{i,j-1,k} - \hat{C}^+_{i,j,k-1} \\ D &= \tilde{\rho}I \\ U &= \tilde{\rho}I + \hat{A}^-_{i+1,j,k} + \hat{B}^-_{i,j+1,k} + \hat{C}^-_{i,j,k+1} \end{aligned} \tag{31}$$

where

$$\tilde{\rho} = \tilde{\rho}(\hat{A}) + \tilde{\rho}(\hat{B}) + \tilde{\rho}(\hat{C}) \tag{32}$$

In the inversion process, $\hat{A}^+_{i-1,j,k}$ is multiplied by $\delta\hat{Q}^*_{i-1,j,k}$, for example. The algorithm permits scalar diagonal inversions since

$$\text{diag}(L \quad \text{or} \quad U) = \begin{bmatrix} \tilde{\rho} & 0 & 0 & 0 & 0 \\ 0 & \tilde{\rho} & 0 & 0 & 0 \\ 0 & 0 & \tilde{\rho} & 0 & 0 \\ 0 & 0 & 0 & \tilde{\rho} & 0 \\ 0 & 0 & 0 & 0 & \tilde{\rho} \end{bmatrix} \tag{33}$$

The use of the true Jacobian matrices of Eq. (23), which might lead to a faster convergence rate, requires block diagonal inversions and hence approximately doubles the computational work per iteration. Another interesting feature of the present algorithm is that the scheme is completely vectorizable on $i + j + k = \text{const}$ oblique planes of sweep, which is illustrated in Fig. 1. This is achieved by reordering the three-dimensional arrays into two-dimensional arrays, i.e.,

$$\hat{Q}(i\text{point}, i\text{plane}) = \hat{Q}(i, j, k) \tag{34}$$

where $i$plane is the serial number of the oblique plane to be swept, and $i$point is the address on that plane. The present algorithm may also be amenable to parallel processing.

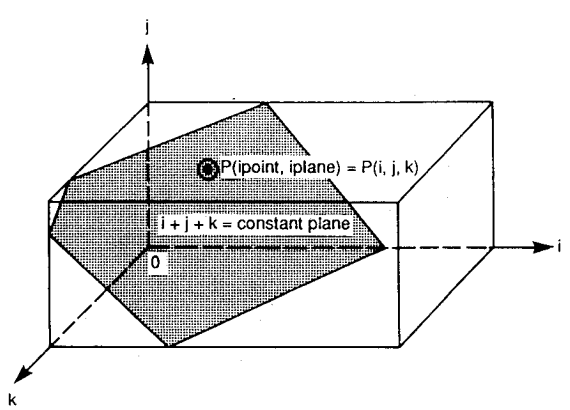

Fig. 1 Oblique plane of sweep for vectorization.

IV. Numerical Dissipation

A semidiscrete finite volume method is used to ensure that the final converged solution be independent of the time step and to avoid metric singularity problems. The finite volume method is based on the local flux balance of each mesh cell. For example,

$$\begin{aligned} \partial_\xi\hat{E} + \partial_\eta\hat{F} + \partial_\zeta\hat{G} &= \hat{E}_{i+\frac{1}{2},j,k} - \hat{E}_{i-\frac{1}{2},j,k} + \hat{F}_{i,j+\frac{1}{2},k} \\ &\quad - \hat{F}_{i,j-\frac{1}{2},k} + \hat{G}_{i,j,k+\frac{1}{2}} - \hat{G}_{i,j,k-\frac{1}{2}} \end{aligned} \tag{35}$$

A central difference scheme achieves the second order accuracy in the most efficient way when the flowfield is free of discontinuous solutions. However, numerical dissipation models are added to nondissipative central difference schemes to suppress the tendency for odd and even point decoupling. Dissipation models are often called "filters" since they work like low pass filters that damp out high-frequency modes. The dissipative flux d is added to the convective flux in a conservative manner:

$$\begin{aligned} &(\hat{E}_{i+\frac{1}{2},j,k} - \hat{E}_{i-\frac{1}{2},j,k} + \hat{F}_{i,j+\frac{1}{2},k} - \hat{F}_{i,j-\frac{1}{2},k} + \hat{G}_{i,j,k+\frac{1}{2}} \\ &\quad - \hat{G}_{i,j,k-\frac{1}{2}}) - (d_{i+\frac{1}{2},j,k} - d_{i-\frac{1}{2},j,k} + d_{i,j+\frac{1}{2},k} - d_{i,j-\frac{1}{2},k} \\ &\quad + d_{i,j,k+\frac{1}{2}} - d_{i,j,k-\frac{1}{2}}) \end{aligned} \tag{36}$$

For simplicity, $d_{i+\frac{1}{2},j,k}$ is denoted by $d_{i+\frac{1}{2}}$ hereafter.

It has long been recognized that characteristic-based upwind-biased schemes can demonstrate crisp resolution of dis-

continuities. This is especially so when the flux-difference splitting scheme replaces Godunov's exact solution of the Riemann problem with an approximate solution, while distinguishing between the influence of forward and backward moving waves. High-order upwind schemes can be constructed by using multipoint extrapolation formulas to estimate the numerical flux, or by adding higher-order dissipative terms. In either case flux limiters are then added to control the signs of the coefficients of a semidiscrete approximation to the hyperbolic system of equations. The dissipative coefficient for a system of equations must be a matrix to meet the requirement of upwinding. It is sometimes necessary to add artificial dissipation in the form of entropy correction to avoid instabilities. Considering the additional cost and reduced robustness of the upwind-biased scheme when the grid lines are not aligned with strong shock waves,[11] it seems that the flux-limited dissipation model with scalar coefficients can be a practical alternative to upwind dissipation with matrix coefficients, especially when the uncertainty of the solution due to a turbulence model is relatively large.

In the flux-limited dissipation model, the dissipative flux is constructed by introducing flux limiters into the high-order terms instead of adding low-order terms:

$$d_{i+\frac{1}{2}} = -\alpha_{i+\frac{1}{2}}[\phi(\sigma_{i+1})e_{i+3/2} - 2e_{i+\frac{1}{2}} + \psi(\sigma_i)e_{i-\frac{1}{2}}] \quad (37)$$

where ϕ and ψ are flux limiting functions to limit antidiffusive fluxes:

$$\phi(\sigma) = \begin{bmatrix} 0 & \text{if} & \sigma < 0 \\ \sigma & \text{if} & 0 \le \sigma \le 1 \\ 1 & \text{if} & \sigma > 1 \end{bmatrix} \quad (38)$$

and

$$\psi(\sigma) = \phi(1/\sigma) \quad (39)$$

Here,

$$\sigma_i = \frac{e_{i-\frac{1}{2}}}{e_{i+\frac{1}{2}}} \quad (40)$$

and

$$e_{i+\frac{1}{2}} = \hat{Q}_{i+1} - \hat{Q}_i \quad (41)$$

If we write $\sigma = b/a$, then

$$\phi(\sigma)a = \text{min-mod}(a, b) \quad (42)$$

where min-mod(a, b) is zero if a and b have opposite signs, and min-mod(a, b) is the smaller of a and b if a and b have the same sign:

$$\alpha_{i+\frac{1}{2}} = (\kappa_0 + \kappa_1\hat{\nu}_{i+\frac{1}{2}})r(\hat{A})_{i+\frac{1}{2}} \quad (43)$$

where the constant κ_0 determines a threshold, and the constant κ_1 is chosen to ensure that there is enough dissipation to suppress numerical oscillations in the neighborhood of shock waves. The expression $r(\hat{A})$ denotes the spectral radius of the Jacobian matrix $\hat{A}$, and $\hat{\nu}_{i+\frac{1}{2}}$ is a sensor:

$$\hat{\nu}_{i+\frac{1}{2}} = \max(\nu_{i+1}, \nu_i) \quad (44)$$

where

$$\nu_i = \max(\nu_i^p, \nu_i^T) \quad (45)$$

$$\nu_i^p = \frac{|p_{i+1} - 2p_i + p_{i-1}|}{(p_{i+1} + 2p_i + p_{i-1})} \quad (46)$$

$$\nu_i^T = \frac{|T_{i+1} - 2T_i + T_{i-1}|}{(T_{i+1} + 2T_i + T_{i-1})} \quad (47)$$

where p and T are the pressure and the temperature, respectively.

V. Results

The LU-SGS algorithm can be completely vectorized, and its efficiency is demonstrated by the CENS3D code on a Cray YMP supercomputer at NASA Ames Research Center. The CENS3D code requires only 9-μs per grid point per iteration for the thin-layer option of the Navier-Stokes equations with an algebraic turbulence model on a single processor at the sustained rate of 175 MFLOPS. Approximately 55, 20, and 20% of the computing time are spent for the implicit matrix operation, the numerical dissipation, and the evaluation of viscous fluxes, respectively. It is interesting to note that the LU-SGS scheme requires less computational work per iteration than some explicit schemes. Based on experience with INS3D-LU,[21] an incompressible flow code that employs the LU-SGS scheme and achieved 1.2 GFLOPS using eight processors,[36] the CENS3D is expected to perform very well on shared-memory multiple processors. In a recent study,[37] the

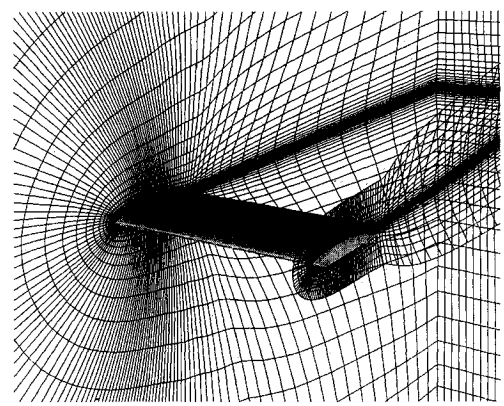

Fig. 2 Computational grid for NACA 64A010 wing.

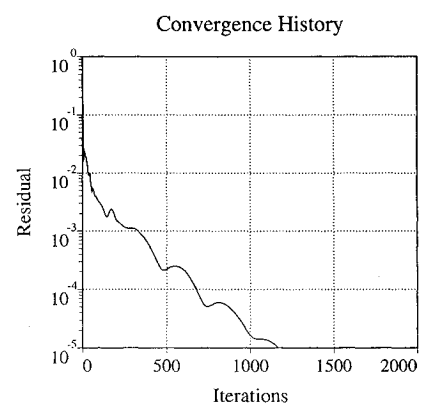

Fig. 3 Convergence history for NACA 64A010 wing.

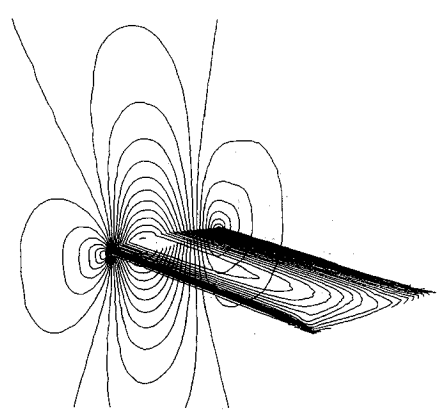

Fig. 4 Pressure contours for NACA 64A010 wing.

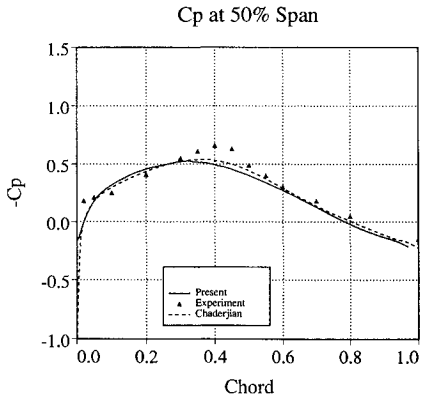

Fig. 5 C_p at 50% span for NACA 64A010 wing.

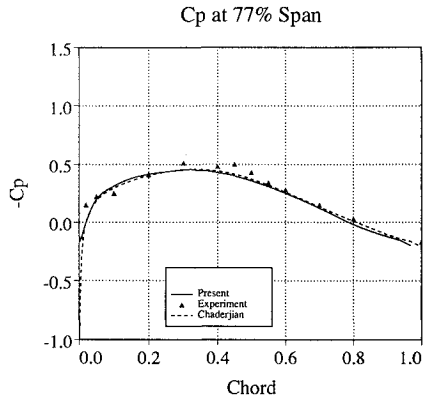

Fig. 6 C_p at 77% span for NACA 64A010 wing.

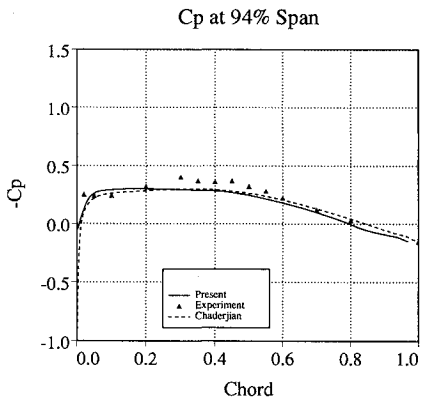

Fig. 7 C_p at 94% span for NACA 64A010 wing.

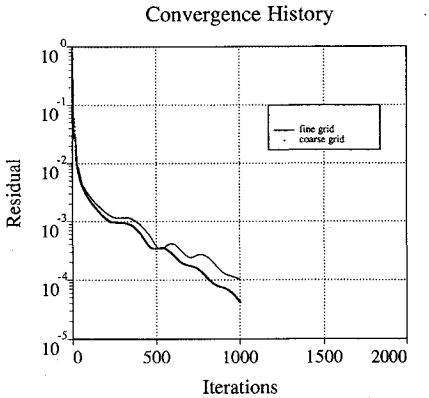

Fig. 8 Convergence histories for ONERA M6 wing.

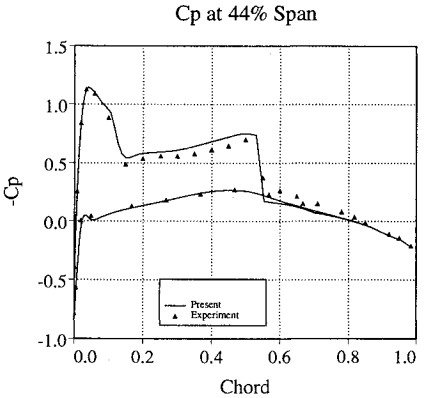

Fig. 9 C_p at 44% span for ONERA M6 wing.

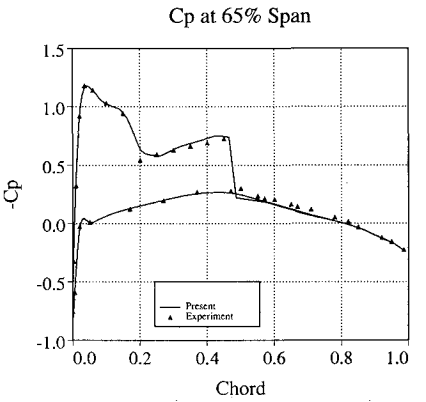

Fig. 10 C_p at 65% span for ONERA M6 wing.

LU-SGS algorithm outperformed the existing implicit schemes on a massively parallel computer such as the Connection Machine CM-2.

To validate the new CENS3D code, calculations have been performed for an NACA 64A010 wing. The thickness-to-chord ratio of the wing, which has an aspect ratio of 4, has been modified to 10.6%. The experiment was conducted by Mabey et al.[38] in the RAE 8 × 8 ft wind tunnel. The model was mounted on a fuselage-like body to displace it slightly from the wind tunnel wall and its boundary layer. However, no attempt has been made here to model the test section. A 151 × 39 × 39 C-H mesh (229,671 points) generated by Chaderjian and Guruswamy[39] is used for the present calculation. Figure 2 shows a partial view of the computational grid. The freestream conditions are Mach 0.8, Reynolds number 2.4×10^6, and 0 deg angle of attack. The algebraic turbulence model by Baldwin and Lomax[40] is employed for mathematical closure of the Reynolds-averaged Navier-Stokes equations. Original coefficients are used except C_{wk}, and the coefficient for F_{wake} is set to 1 instead of 0.25, as done in Ref. 39. The y^+ values at the first mesh cell, which are adjacent to the wing surface near the midspan, are ~ 2. The convergence history in Fig. 3 shows that the root-mean-squared residual of the continuity equation drops three orders of magnitude in about 340 iterations, or 12 CPU minutes. Pressure contours are shown in Fig. 4. The computed pressure coefficients are compared with experimental data and the numerical solution of Chaderjian and Guruswamy[39] in Figs. 5–7. Their code uses a finite difference discretization, artificial dissipation using blended second and fourth differences, a diagonalized ADI scheme, and the Baldwin-Lomax turbulence model. Figures 5–7 correspond to C_p comparisons at 50, 77, and 94% semispan stations, respectively. Values at the leading and trailing edges are not available for plotting because flow variables are located at cell centers. Overall agreement between the two numerical solutions is seen to be good despite the differences in numerical formulation. The slight discrepancy between the experimental data and the numerical solutions may be due to effects of the fuselage-like body at the wing root and the wind tunnel wall that are not modeled in the numerical simulations.

For additional validation of the code, transonic flow calculations have been carried out for an ONERA M6 wing. A $289 \times 50 \times 44$ C-H mesh (635,800 points) is used as a fine grid. The distance of the first grid point from the wing surface is 1.0×10^{-5} chord length of the root section. The freestream conditions are Mach 0.8395, Reynolds number 1.5×10^7, and 3.06 deg angle of attack. The Baldwin-Lomax turbulence model is used again for the attached flow simulation. The residual drops to three orders in about 380 iterations, or 38 CPU minutes on the fine grid. In the present implementation, implicit viscous terms are not included to avoid the increase of computational work per iteration. To investigate the effect of this compromise on the convergence rate, a grid-convergence study has been performed. Figure 8 shows the convergence histories on both a fine grid and a $171 \times 25 \times 44$ (188,100 points) coarse grid. Although the number of grid points needed to resolve the viscous boundary layer is doubled, the convergence is seen to be slowed by only 20%. Figures 9 and 10 show a good agreement between experimental data[41] and the pressure coefficients at 44 and 65% semispan stations computed on the fine grid.

Conclusions

A three-dimensional numerical method based on the LU-SGS implicit scheme in conjunction with the flux-limited dissipation model is developed for simulating viscous turbulent compressible flows. Good performance of the new testbed code is demonstrated on a Cray YMP computer. Despite its reasonably fast convergence, the LU-SGS scheme requires very low computational time per iteration. The present three-dimensional Navier-Stokes solution of a high Reynolds number flow using 636K grid points is obtained in 38 min.

References

[1]Yoon, S., and Jameson, A., "Lower-Upper Symmetric-Gauss-Seidel Method for the Euler and Navier-Stokes Equations," *AIAA Journal,* Vol. 26, No. 9, 1988, pp. 1025-1026; also AIAA Paper 87-0600, Jan. 1987.

[2]Yoon, S., "Numerical Solution of the Euler Equations by Implicit Schemes with Multiple Grids," Princeton Univ., Mechanical and Aerospace Engineering Rept. 1720-T, Princeton, NJ, Sept. 1985.

[3]Jameson, A., and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal,* Vol. 25, No. 7, 1987, pp. 929-935.

[4]Shuen, J. S., and Yoon, S., "A Numerical Study of Chemically Reacting Flows Using a Lower-Upper Symmetric Successive Overrelaxation Scheme," *AIAA Journal,* Vol. 27, No. 12, 1989, pp. 1752-1760.

[5]Choo, Y. K., Soh, W. Y., and Yoon, S., "Application of a Lower-Upper Implicit Scheme and an Interactive Grid Generation for Turbomachinery Flow Field Simulations," American Society of Mechanical Engineers, New York, Paper 89-GT-20, June 1989.

[6]Yoon, S., and Jameson, A., "Lower-Upper Implicit Scheme for High-Speed Inlet Analysis," *AIAA Journal,* Vol. 25, No. 8, 1987, pp. 1052-1053.

[7]Rieger, H., and Jameson, A., "Solution of Steady Three-Dimensional Compressible Euler and Navier-Stokes Equations by an Implicit LU Scheme," AIAA Paper 88-0619, Jan. 1988.

[8]Yu, S. T., Tsai, Y. L. P., and Shuen, J. S., "Three-Dimensional Calculation of Supersonic Reacting Flows Using an LU Scheme," AIAA Paper 89-0391, Jan. 1989.

[9]Coirier, W. J., "High Speed Corner and Gap Seal Computations Using an LU-SGS Scheme," AIAA Paper 89-2669, July 1989.

[10]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes," AIAA Paper 81-1259, July 1981.

[11]Yoon, S., and Kwak, D., "Artificial Dissipation Models for Hypersonic External Flow," AIAA Paper 88-3708, July 1988.

[12]Yoon, S., and Kwak, D., "Artificial Dissipation Models for Hypersonic Internal Flow," AIAA Paper 88-3277, July 1988.

[13]Obayashi, S., "Numerical Simulation of Underexpanded Plumes Using Upwind Algorithms," AIAA Paper 88-4360-CP, Aug. 1988.

[14]Chen, C. L., McCrosky, W. J., and Obayashi, S., "Numerical Solutions of Forward-Flight Rotor Flow Using an Upwind Method," AIAA Paper 89-1846, June 1989.

[15]Loh, H. T., and Golafshani, M., "Computation of Viscous Chemically Reacting Flows in Hybrid Rocket Motors Using an Upwind LU-SSOR Scheme," AIAA Paper 90-1570, June 1990.

[16]Yungster, S., "Numerical Study of Shock-Wave/Boundary Layer Interactions in Premixed Hydrogen-Air Hypersonic Flows," AIAA Paper 91-0413, Jan. 1991.

[17]Imlay, S. T., and Eberhardt, S., "Nonequilibrium Thermo-Chemical Calculations Using a Diagonal Implicit Scheme," AIAA Paper 91-0468, Jan. 1991.

[18]Park, C., and Yoon, S., "Calculation of Real-Gas Effects on Blunt-Body Trim Angles," *AIAA Journal,* Vol. 30, No. 4, 1992, pp. 999-1007.

[19]Park, C., and Yoon, S., "A Fully-Coupled Implicit Method for Thermo-Chemical Nonequilibrium Air at Sub-Orbital Flight Speeds," *Journal of Spacecraft and Rockets,* Vol. 28, No. 1, 1991, pp. 31-39.

[20]Park, C., and Yoon, S., "Calculation of Real Gas Effects on Airfoil Aerodynamic Characteristics," AIAA Paper 90-1712, June 1990.

[21]Yoon, S., Kwak, D., and Chang, L., "LU-SGS Implicit Algorithm for Three-Dimensional Incompressible Navier-Stokes Equations with Source Term," AIAA Paper 89-1964-CP, June 1989.

[22]Giles, M., Drela, M., and Thompkins, W. T., "Newton Solution of Direct and Inverse Transonic Euler Equations," AIAA Paper 85-1530-CP, 1985.

[23]Beam, R., and Warming, R. F., "An Implicit Factored Scheme for the Compressible Navier-Stokes Equations," *AIAA Journal,* Vol. 16, No. 4, 1978, pp. 393-402.

[24]Briley, W. R., and McDonald, H., "Solution of the Multidimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method," *Journal of Computational Physics,* Vol. 24, No. 4, 1977, pp. 372-397.

[25]Pulliam, T. H., and Chaussee, D. S., "A Diagonal Form of an Implicit Approximate Factorization Algorithm," *Journal of Computational Physics,* Vol. 39, No. 2, 1981, pp. 347-363.

[26]Obayashi, S., and Kuwahara, K., "LU Factorization of an Implicit Scheme for the Compressible Navier-Stokes Equations," *Journal of Computational Physics,* Vol. 63, No. 1, 1986, pp. 157-167.

[27]Steger, J. L., and Warming, R. F., "Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite Difference Methods," *Journal of Computational Physics,* Vol. 40, No. 2, 1981, pp. 263-293.

[28]Ying, S. X., Steger, J. L., Schiff, L. B., and Baganoff, D., "Numerical Simulation of Unsteady, Viscous, High Angle-of-Attack Flows Using a Partially Flux Split Algorithm," AIAA Paper 86-2179, Aug. 1986.

[29]Rizk, Y. M., Chaussee, D. S., and Steger, J. L., "Numerical Simulation of the Hypersonic Flow Around Lifting Vehicles," NASA TM-89444, 1987.

[30]Edwards, T. A., and Flores, J., "Toward a CFD Nose-to-Tail Capability: Hypersonic Unsteady Navier-Stokes Code Validation," AIAA Paper 89-1672, June 1989.

[31]Jameson, A., and Turkel, E., "Implicit Schemes and LU Decompositions," *Mathematics of Computation,* Vol. 37, No. 156, 1981, pp. 385-397.

[32]Buning, P. G., and Steger, J. L., "Solution of the Two-Dimensional Euler Equations with Generalized Coordinate Transformation Using Flux Vector Splitting," AIAA Paper 82-0971, June 1982.

[33]Whitfield, D. L., "Implicit Upwind Finite Volume Scheme for the Three-Dimensional Euler Equations," Mississippi State Univ. Rept. MSSU-EIRS-ASE-85-1, Mississippi State, MS, Sept. 1985.

[34]Buratynski, E. K., and Caughey, D. A., "An Implicit LU Scheme for the Euler Equations Applied to Arbitrary Cascades," AIAA Paper 84-0167, Jan. 1984.

[35]MacCormack, R. W., "Current Status of Numerical Solutions of the Navier-Stokes Equations," AIAA Paper 85-0032, 1985.

[36]Fatoohi, R., and Yoon, S., "Multitasking the INS3D-LU Code on the Cray Y-MP," AIAA Paper 91-1581, June 1991, 1992.

[37]Fatoohi, R., private communication, 1992.

[38]Mabey. D. G., Welsh, B. L., and Pyne, C. R., "A Summary of Measurements of Steady and Oscillatory Pressures on a Rectangular Wing," *Aeronautical Journal of the Royal Aeronautical Society,* Vol. 92, No. 911, Jan. 1988, pp. 10-28.

[39]Chaderjian, N. M., and Guruswamy, G. P., "Unsteady Transonic Navier-Stokes Computations for an Oscillating Wing Using Single and Multiple Zones," AIAA Paper 90-0313, Jan. 1990.

[40]Baldwin, B. S., and Lomax, H., "Thin-Layer Approximation and Algebraic Model for Separated Turbulent Flow," AIAA Paper 78-0257, Jan. 1978.

[41]Schmitt, V., and Charpin, F., "Pressure Distributions on the ONERA M6 Wing at Transonic Mach Numbers," AGARD AR-138-B1, 1979.